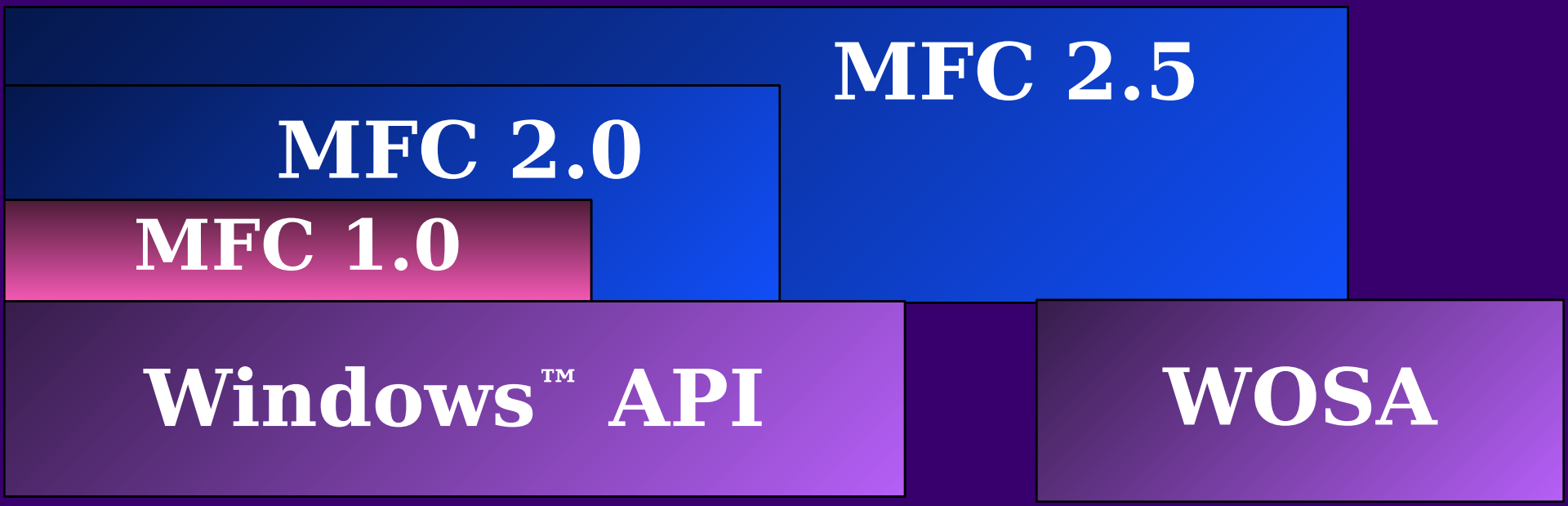
Microsoft® Foundation Classes Architecture Overview

Greg DeMichille
Lead Developer
AFX Group
Microsoft Corporation

MFC Architecture

- MFC evolution
- MFC overview
- Portability
- Architecture classes
- High-level abstractions
- OLE 2.0 specifics

MFC Evolution



MFC Evolution

- MFC 1.0: April ’92
  - General-purpose classes
  - Windows API classes
- MFC 2.0: February ’93
  - Architecture classes
  - High-level abstractions
- MFC 2.5: December ’93
  - Database classes

MFC 1.0 Overview

- Utilizes *sane* subset of C++
- Provides the C++ API to Windows
- Adhere to standard user-interface idioms, but enable customization
- Compiler-independent
- Results in *small/fast* executables
- Recompiles to Win32s™, Windows NT™

MFC 2.0 Overview

- Builds on *MFC 1.0*
  - The C++ interface to Windows
  - Backwardly compatible
- Implements many features based on suggestions/input from developers
- 100+ classes, 60,000 lines of code
- Works with Visual C++™ wizards and AppStudio

MFC 2.5 Overview

- Builds on *MFC 2.0*
  - Accepted standard
  - Backwardly compatible
- Bugs fixes, enabling features, DBCS
- Database through ODBC
- OLE 2.0
- Works with Visual C++ 1.5 wizards

General-Purpose Classes

- Run-time type information
- Object persistence
- Data structures/collections
- Strings
- Files
- Time and date
- Exception handling

Windows API Classes

- Standard application support
- Window management
- Frame windows, MDI
- Graphics/GDI
- Menus
- Controls
- Dialogs

Architecture Classes

- Documents and views
- Commands and command routing
- Printing and print preview
- Dialog data exchange and validation (DDX/DDV)
- Database engine classes
- Context-sensitive Help

High-Level Abstractions

- Toolbar and other control bars
- Scrolling view and splitter window
- Form view, record view
- Edit view
- VBX controls (16 bits only)
- OLE 2.0 support

OLE 2.0 Classes

- OLE Automation client
- OLE Automation server
- OLE Visual Editing container
- OLE Visual Editing server
- OLE Visual Editing support

Size And Speed

- Rebuilt MFC 2.x executables only slightly larger than 1.0
- MFC 2.x *MultiPad*
  - 87 lines (C is 2315, MFC 1.0 is 1929)
  - And it has many *more* features
- No speed penalty and some cases are faster using message maps
- Shared DLLs as packing option

Shipping

- MFC 2.0 shipping now:
  - Visual C++ 1.0 Standard and Professional Edititons (16-bit)
  - Visual C++ 1.0 32-bit Edition
  - Symantec C++ 6.0 (16-bit)
- Coming soon:
  - MFC 2.5 with Visual C++ 1.5 (16-bit)
  - MFC 3.0 with Visual C++ 2.0 (32-bit)
  - And MFC 3.0 on Mac®, MIPS, Alpha...

Shipping With Visual C++

- Complete API reference
- Tutorial, cookbook/encyclopedia
- Technotes, sample Help
- Sample source code (~30,000 lines)
- Library source code (~60,000 lines)
- Visual C++ tools and wizards

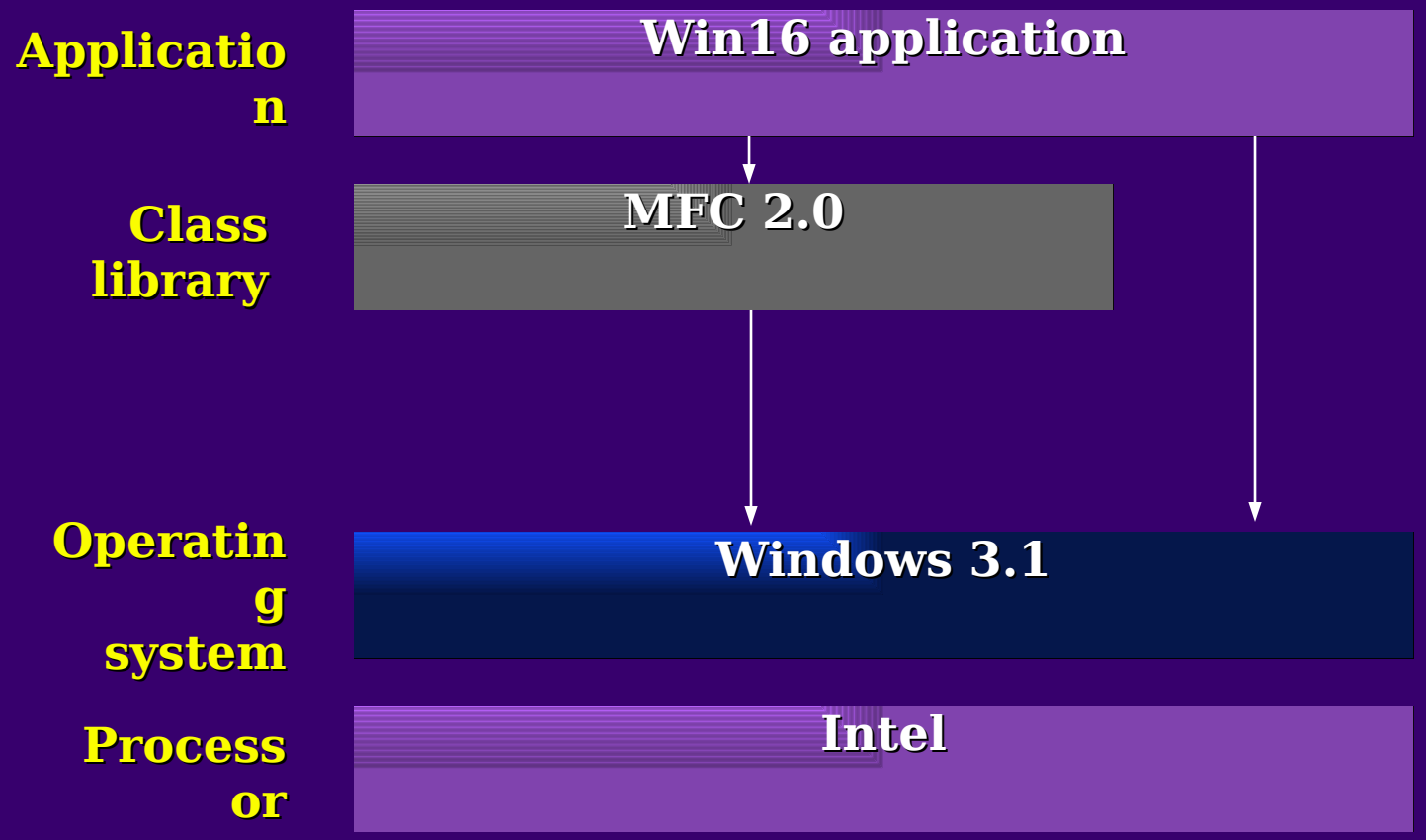
Portability Strategy

- Microsoft strategy
  - Multiplatform tools
  - Windows API on many platforms
- Your code
  - MFC for C++ code
  - Windows API for C code
  - 16-bit and 32-bit
  - Source code recompile

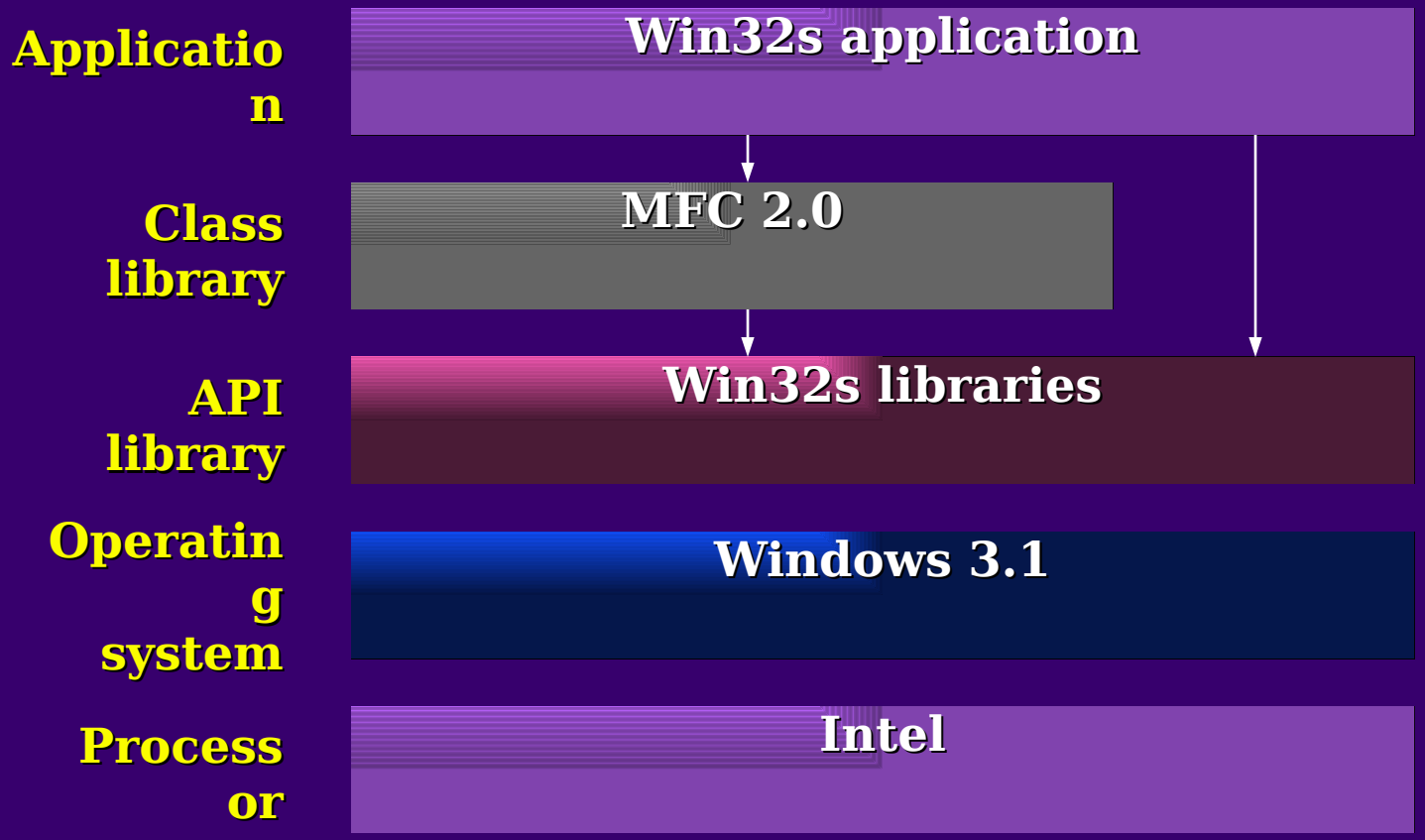
Platforms

- Windows platforms
  - Windows 3.1
  - Win32 and Win32s™ Intel
  - Win32 RISC
- Platforms other than Windows
  - Macintosh

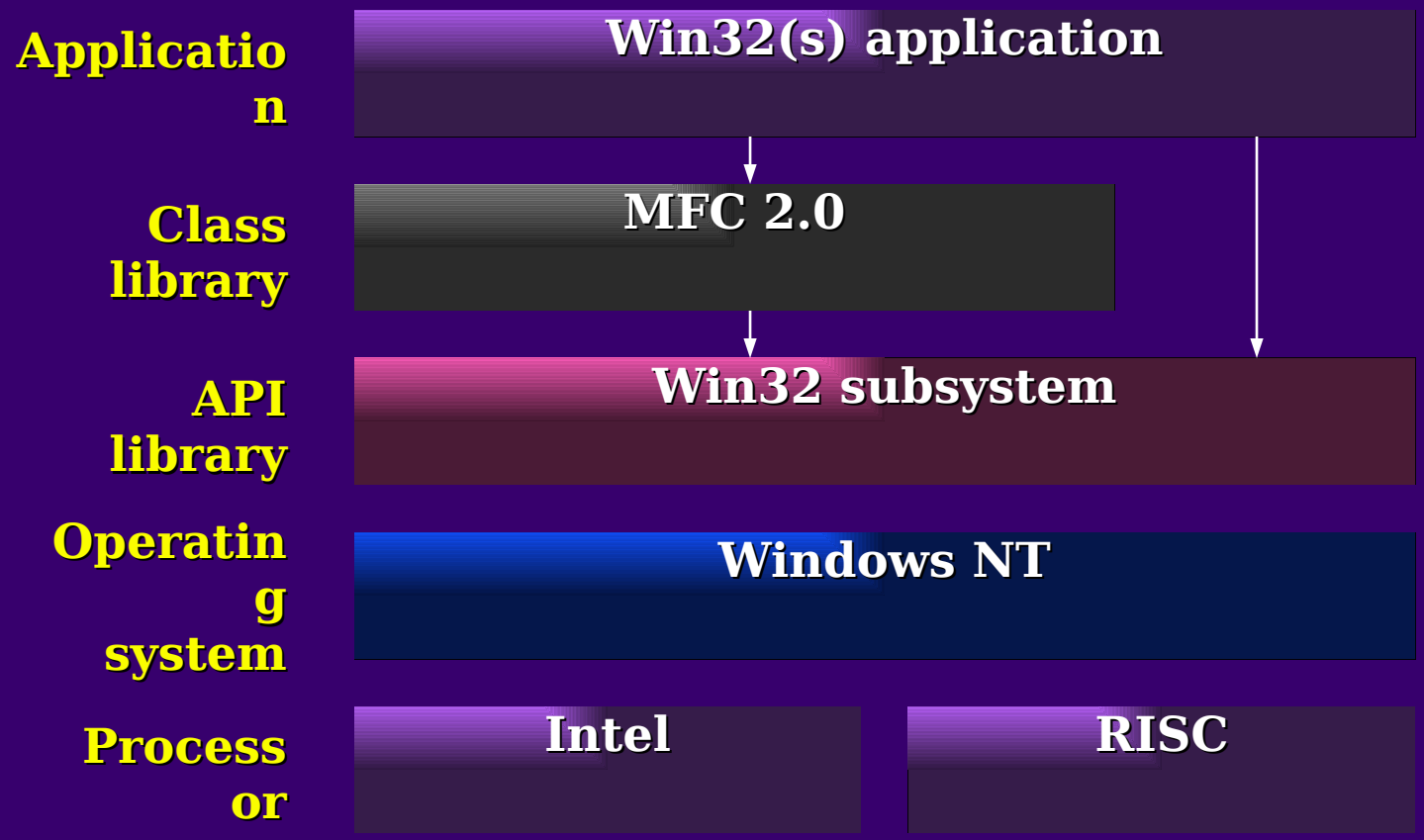
Win16 Architecture



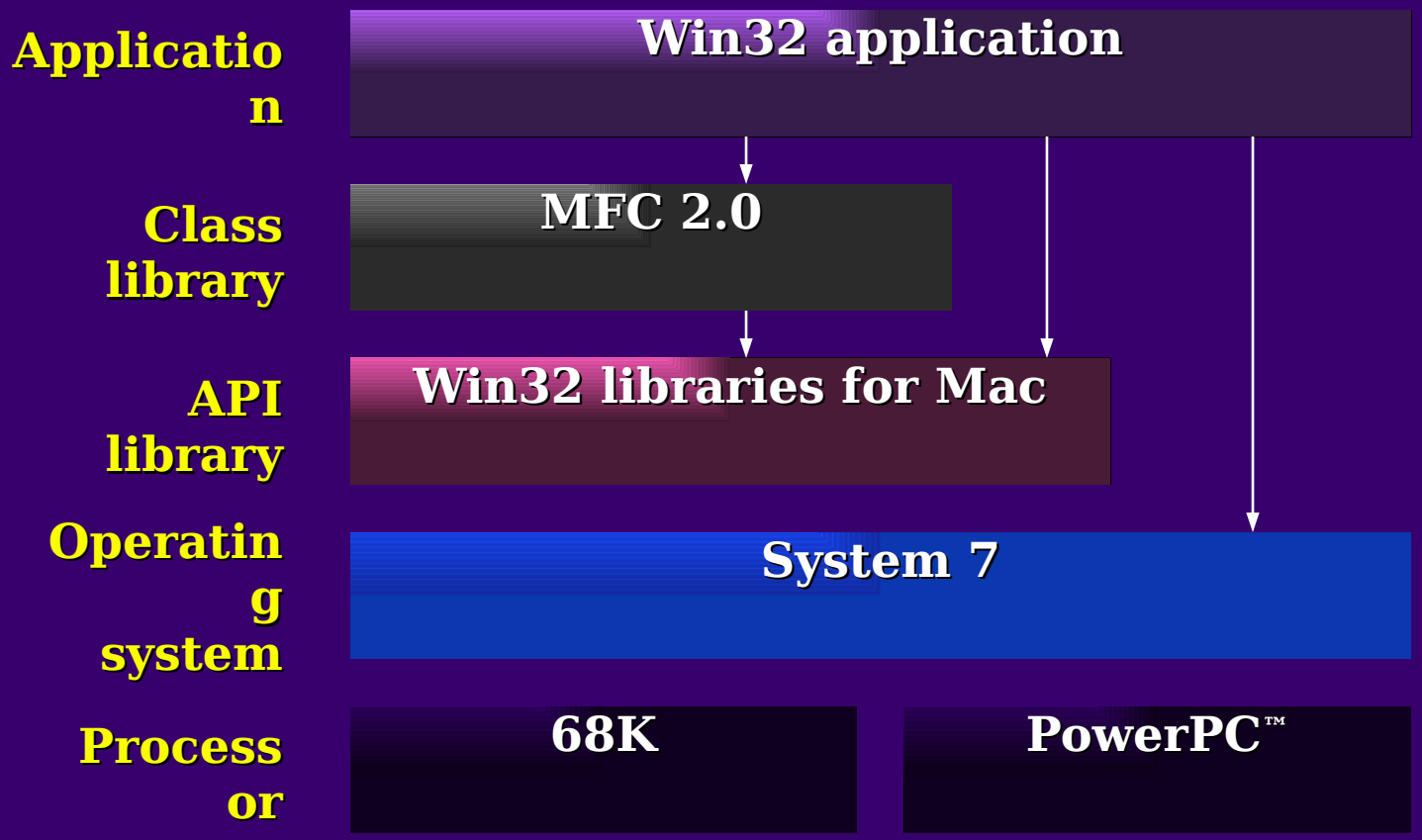
Win32s Architecture



Win32 Architecture



Application Architecture For Mac



Architecture Classes

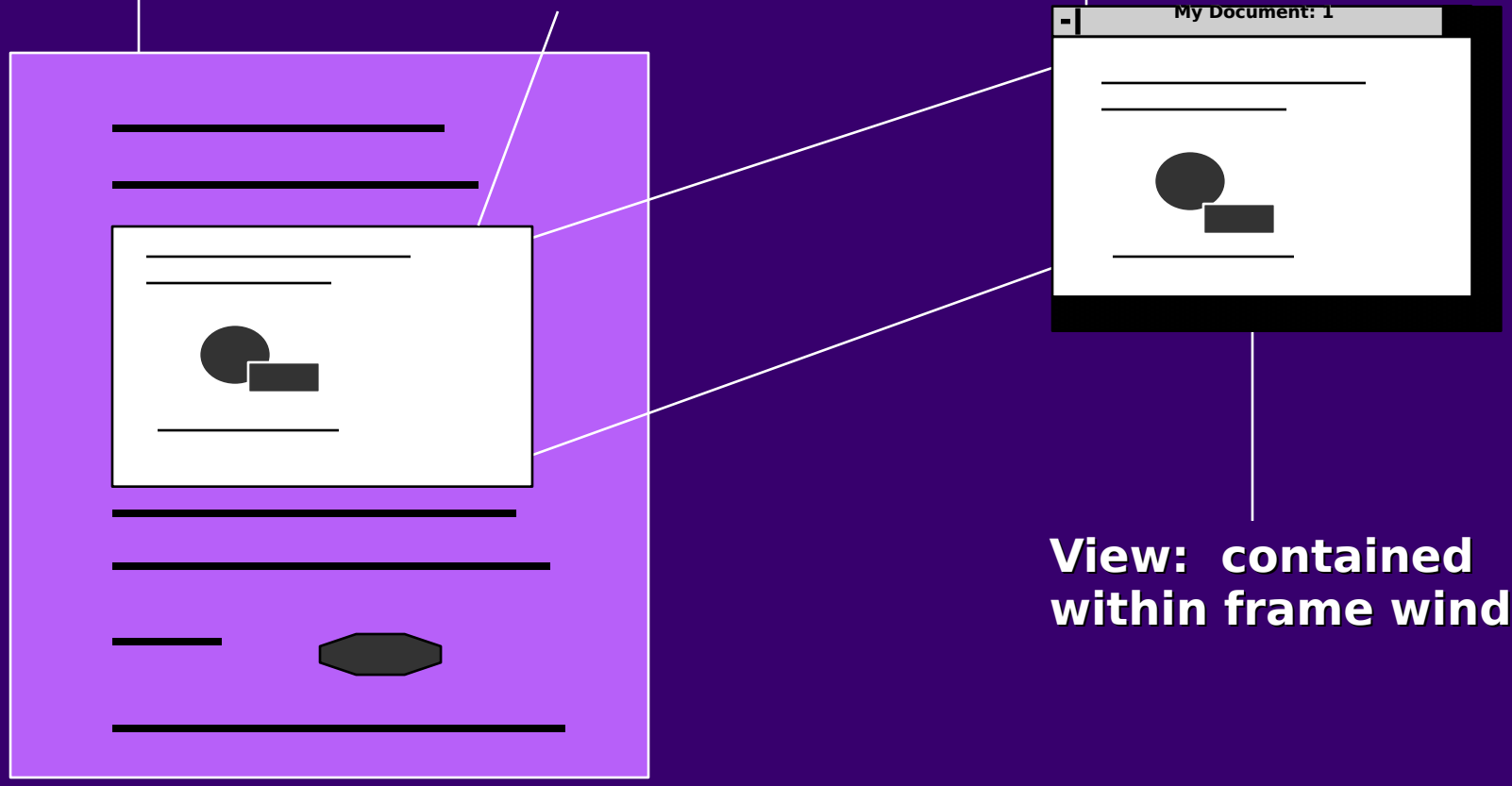
- Documents and views
- Commands and command routing
- Printing and print preview
- Dialog data exchange and validation (DDX/DDV)
- Database engine classes
- Context-sensitive Help

Documents And Views

Document:
stores data in
efficient format

Visible
portion of
document

View: renders visible
data and responds to
user



View: contained
within frame window

Documents And Views

- CDocument
  - Manages application-specific data
  - Supports disk file, database, OLE Compound File, non-file-based storage mechanisms
  - Loads and saves state by overriding `Serialize` member function
  - MFC handles the standard UI

Documents And Views

- `CView`
  - CView *is-a* simple child window
  - Can be anywhere (SDI, MDI, in-place)
  - Supports multiple (distinct) views on a single document
  - Replaces low-level `OnPaint` with abstract `OnDraw` overridable
  - Provides change notification and enables optimized drawing

Documents And Views

Example:
SCRIBBLE

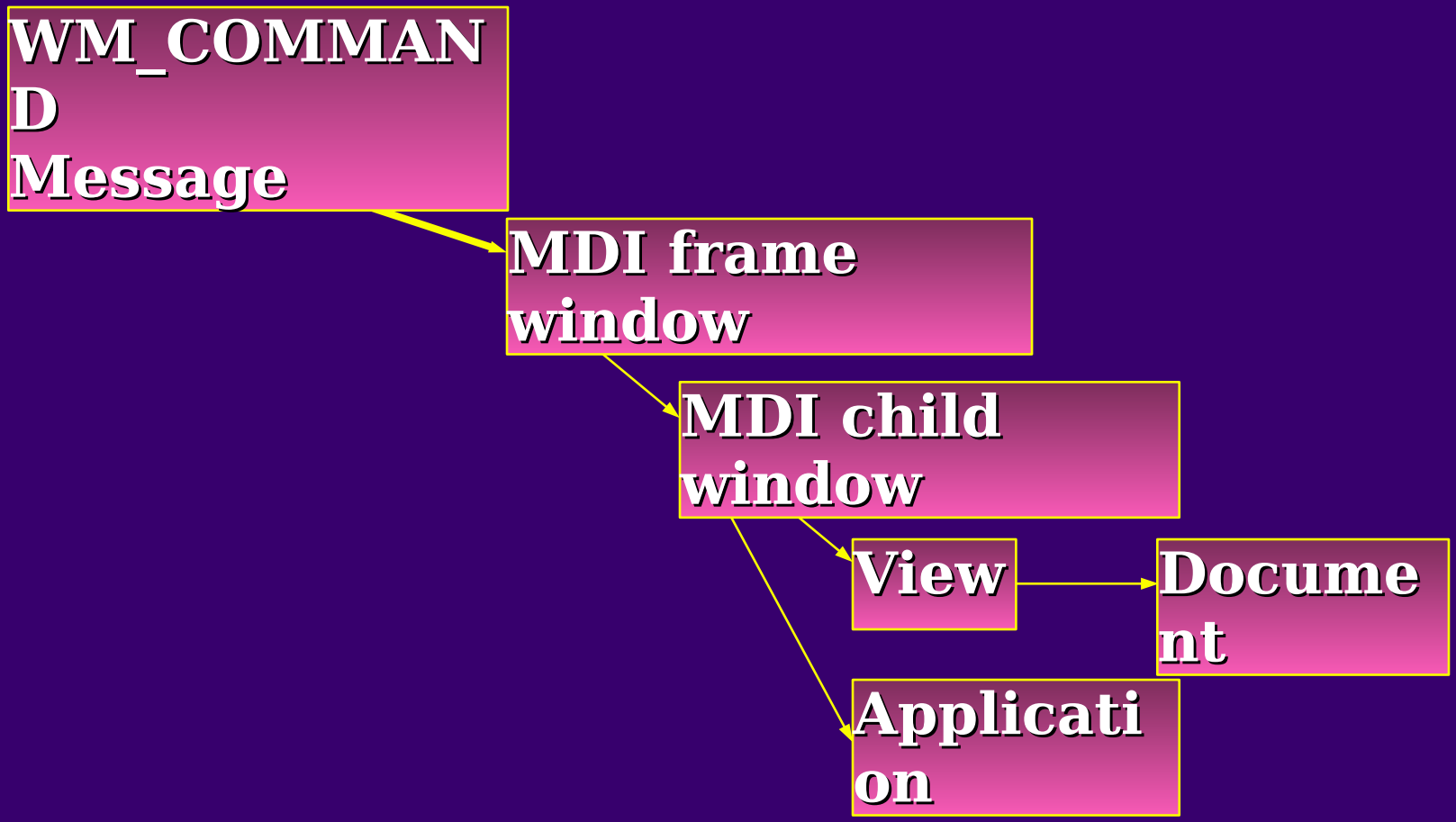
Architecture Classes

- Documents and views
- Commands and command Routing
- Printing and print preview
- Dialog data exchange and validation (DDX/DDV)
- Database engine classes
- Context-sensitive Help

Commands

- Instructed to perform an action
- Built upon MFC 1.0 *message maps*
- Based on WM_COMMAND
- Update command UI updates:
  - Menu items
  - Toolbar buttons
  - Button

Command Routing



Commands And Routing

Example:

SCRIBBLE

Architecture Classes

- Documents and views
- Commands and command Routing
- Printing and print preview
- Dialog data exchange and validation (DDX/DDV)
- Database engine classes
- Context-sensitive Help

Printing

- Leverages doc/view architecture
- Implements device-independent printing
- Provides full-featured print preview
- Uses OnDraw to transparently share rendering code

```
void CMyView::OnDraw(CDC* pDC)
  {
    pDC->TextOut(...);
  }
```

Print And Print Preview

Example:

SCRIBBLE

Architecture Classes

- Documents and views
- Commands and command Routing
- Printing and print preview
- Dialog data exchange and validation (DDX/DDV)
- Database engine classes
- Context-sensitive Help

DDX And DDV

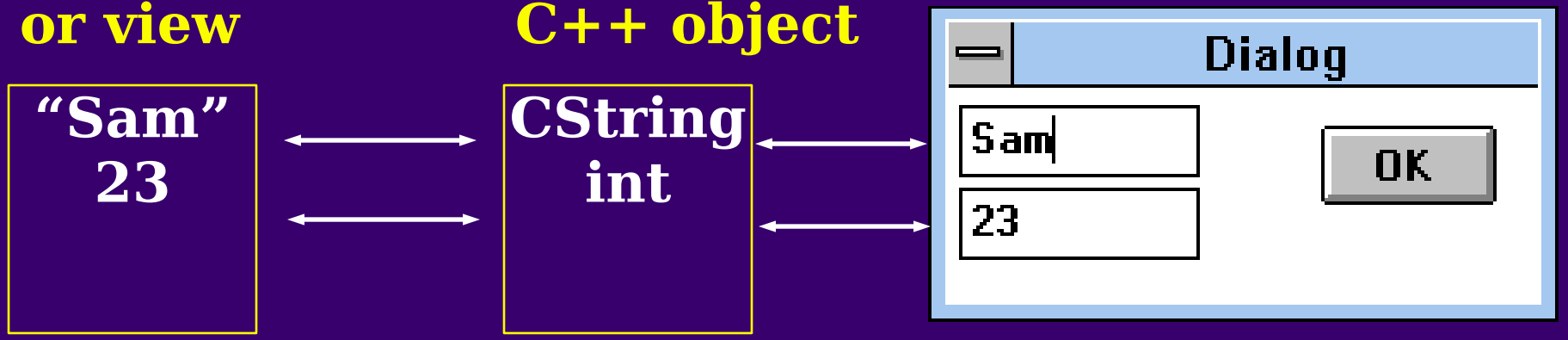
- Initializes, validates, and obtains data values from dialog controls
- Encapsulates dialog code to maximize reusability of a dialog
- Extensible by writing custom exchange and validation routines
- Dialogs, forms, record-based forms

Dialog Model

Document or view

Dialog or form C++ object

Actual dialog or form




```
dlg.m_name =
"Sam";
dlg.m_age = 23;
```

Automatic DDX

Architecture Classes

- Documents and views
- Commands and command routing
- Printing and print preview
- Dialog data exchange and validation (DDX/DDV)
- Database engine classes
- Context-sensitive Help

Database Engine Classes

- Classes *similar* to data objects found in Visual Basic® and Microsoft Access®
- `CDatabase`
  - A *data source* connection
  - Database format independent (ODBC)
- `CRecordset`
  - A scrollable *cursor*
  - C++ data binding with no coding
  - ClassWizard support

DDX, DDV, And Database

Example:
ENROLL

High-Level Abstractions

- Toolbar and other control bars
- Scrolling view and splitter window
- Form view, record view
- Edit view
- VBX controls (16 bits only)
- OLE 2.0 support

Toolbar

- Requested by many MFC 1.0 users
- Implements standard look and feel of Windows
- Shares images from one bitmap edited with AppStudio
- Scalable to toolbar for “Chicago”

Control Bars

Example:

CTRLBARS

High-Level Abstractions

- Toolbar and other control bars
- Scrolling view and splitter window
- Form view, record view
- Edit view
- VBX controls (16 bits only)
- OLE 2.0 support

Scrolling View

- Scrolling display of a portion of document
- Your view derived from `CScrollView`
- Manages window sizes, mapping mode, UI interactions
- Manages OLE in-place scrolling

Splitter Window

- Split window into two or more separately scrollable views
- `CSplitterWnd` manages panes and UI interactions
- Supports dynamic (user-created) or static (application-created) splits

Scrolling And Splitting

Example:

SCRIBBLE

High-Level Abstractions

- Toolbar and other control bars
- Scrolling view and splitter window
- Form view, record view
- Edit view
- VBX controls (16 bits only)
- OLE 2.0 support

Form View

- Uses standard dialog template edited with AppStudio
- Provides power of a dialog in standard view window
  - Scrolling
  - Multiple forms on same data
  - MDI support

Record View

- Derived from `CFormView`
- Tied in to `CRecordset`
- Canned UI for standard forms
  - Next, prev, first, last
- Customizable for more advanced forms

High-Level Abstractions

- Toolbar and other control bars
- Scrolling view and splitter window
- Form view, record view
- Edit view
- VBX controls (16 bits only)
- OLE 2.0 support

OLE 2.0 Classes

- OLE Automation client
- OLE Automation server
- OLE Visual Editing container
- OLE Visual Editing server
- OLE Visual Editing support

OLE Automation Client

- For driving OLE Automation server
  - Server must provide a type library
  - For example, Microsoft Excel 5.0
  - Your code is hybrid C++ and Visual Basic® for Applications
- `COleDispatchDriver`
  - You don’t use it directly
  - ClassWizard will derive classes for you

OLE Automation Server

- Any of your `CCmdTarget` classes
  - Usually document class or
  - Proxy for existing class/code
- Automation requests turn into C++ members through dispatch maps
  - Methods: member functions with C++ types
  - Properties: member variable or call get/set handlers

OLE Automation Server

Example:

AUTOCLIK

OLE Visual Editing Container

- COleDocument
  - Acts as container for COleClientItems
  - Enables structured storage
- COleLinkingDoc
  - Adds support for links to embeddings
- COleClientItem
  - Implements sites and sink interfaces

OLE Visual Editing Server

- COleServerDoc
  - Enables In-Place activation
- COleServerItem
  - Provides glue to your document data
  - Enables links to parts of a document
- COleTemplateServer
  - Handles application launching, doc creation
- COleIPFrameWnd
  - Used for in-place activated servers

OLE Visual Editing Support

- Data Transfer (IDataObject)
  - class `CDataObject` for *using*
  - class `CDataSource` for *implementing*, provided by `COleServerItem`
- Drag-and-Drop feature
  - `COleDropSource`
  - `COleDropTarget`

OLE Visual Editing

Example:

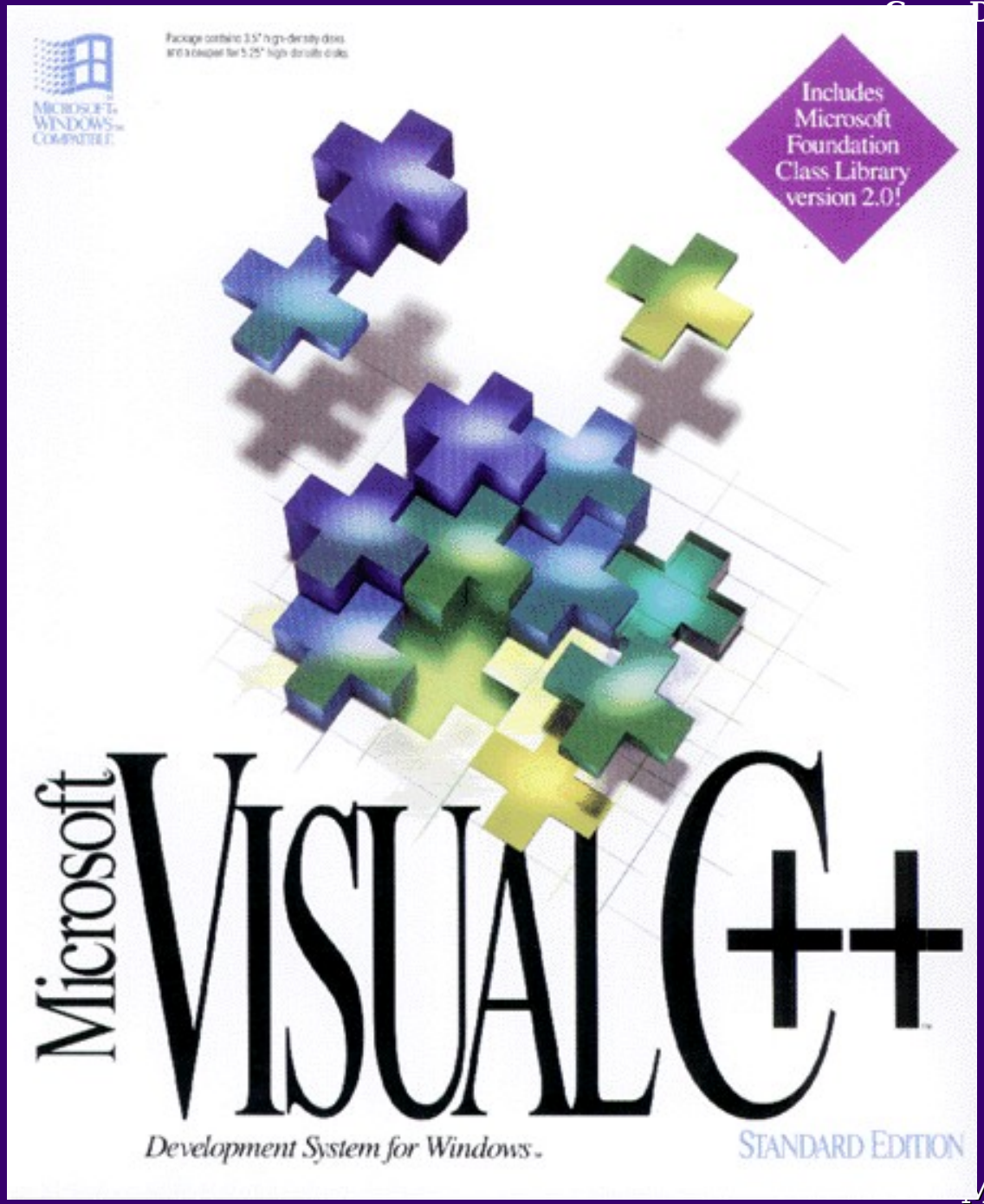
- **SCRIBBLE - the server**
- **DRAWCLI - the container**

Recap

- Architecture classes
  - Infrastructure for your application and rest of the framework
  - Designed for efficiency, generality, and extensibility
- High-level abstractions
  - Builds on the architecture classes
  - Building blocks for your application
  - Designed for feature-rich canned functionality

Summary

- MFC is an integral part of Visual C++
- MFC is *the* C++ Windows API
- MFC
  - Supports an extensive *architecture*
  - Provides *high-level abstractions*
  - Gives portability
  - Results in *small/fast* executables
  - Protects your long-term investment


Includes Microsoft Foundation Class Library version 2.0!
Microsoft
VISUAL C++
Development System for Windows
STANDARD EDITION